August 29, 2005

Intelligent Vehicle Safety Technologies 1 Technical Description

Team Contact Information:
William Klarquist
IVST1 Team Leader
7905 West Stene Drive
Littleton, CO 80128
william.klarquist@perceptek-robotics.com

James McBride
IVST1 Alternate Team Leader
1647 Windy Ridge Court
Saline, MI 48176
jmcbride@ford.com

Prepared for:

DARPA Grand Challenge 2005
THE DEFENSE ADVANCED
RESEARCH PROJECTS AGENCY
3701 NORTH FAIRFAX DRIVE
ARLINGTON, VA 22203-1714

Abstract

The Intelligent Vehicle Safety Technologies 1 team has designed and implemented an advanced autonomous ground platform using a combination of state-of-the-art navigation, laser, vision, radar and ultrasonic sensing to provide input to a software architecture which intelligently incorporates the correct modalities for efficient and effective perceptive planning using environmental contexts. These contexts cover a discrete set of environmental conditions and are selected based on analysis of the a priori path and sensor feedback. The Desert Tortoise vehicle platform is a Ford F250 SuperDuty truck with modifications to provide improved desert performance and environmental protection against shock, high temperature, and dust.

1 Team Introduction

The Intelligent Vehicle Safety Technologies 1 (IVST1) team has been built as a collaboration among a wide array of technical experts in vehicle platforms, navigation, path planning, sensors, perception, and intelligent autonomous control for ground vehicles. The goal has been to both gain insight into domains beyond our individual expertise as well as come together to build a robust autonomous platform fusing these domains. The core team draws on sponsorship from Ford, Honeywell, Delphi, and PercepTek and the members share a common goal: to create safer vehicle systems through new concepts in sensing and perception and the evolution of autonomous navigation and control technologies.

2 Vehicle Description

2.1 Base Platform

The base platform for this effort is a 2005 Ford F250 SuperDuty truck with a 6.0L V8 PowerStroke TurboDiesel Engine. The F250 truck series has been extensively used in desert racing and the inherent mobility provided by this platform is critical to ensuring a capable system. Figure 1 shows the platform and the associated platform dimensions.

2.2 Modifications to Enhance Inherent Mobility

The stock F250 was enhanced both through the use of off-the-shelf components and custom modifications. The off-the-shelf components are BFGoodrich Baja T/A[KR] tires which have been a standard for desert racing and Tractech high performance locking differentials, Truetrac on the front axle and Detroit Locker on the rear axle to provide improved four-wheel-drive performance. The custom modifications were provided by Poison Spyder Customs and include a front sensor rack, skid plates to protect the transmission and differentials, and side tubular bars for additional side protection.

2.3 Actuation

The F250 was also modified to provide actuation of steering, throttle, brake, and transmission. The actuation packages are designed to allow both autonomous control as well as human driving with a set of switches to engage and disengage the actuation.

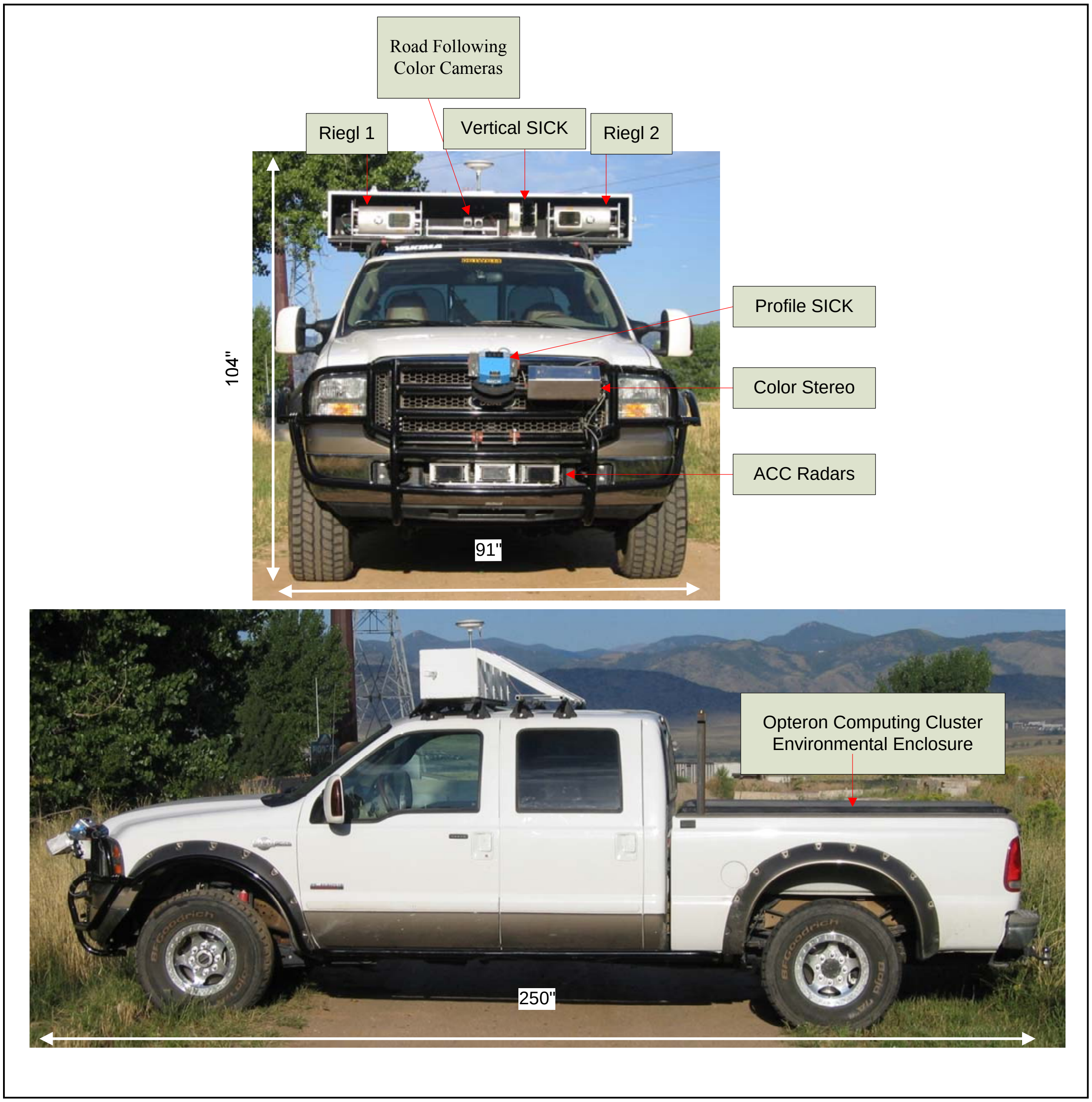


Figure 1: F250 Platform.

2.3.1 Steering

A permanent magnet DC motor coupled to the steering column is used with the stock power steering unit to provide steering control through a Pulse Width Modulated (PWM) interface.

2.3.2 Throttle

The production F250 throttle-by-wire capability is used with a microcontroller interface.

2.3.3 Brake

An advanced brake development system utilizing stock braking system is used to control braking.

2.3.4 Transmission

The automatic transmission is actuated by a linear actuator connected through a cable to the transmission. In order to maintain shift capability independent of computer control, this actuator is controlled by a PIC18F4550 microprocessor which accepts commands from both the computing system and manual control panel.

2.4 Power System

The power system uses both the stock F250 alternator and a Leese Neville 350A alternator to provide the power needed for actuation, sensing, and computing onboard the F250. A Xantrex Prosine Sine Wave Inverter/Charger 3.0 is used for power conditioning, battery charging, and supply of AC power to the computing cluster. The system provides 3000W of continuous power to the other subsystems. A bank of four deep cycle Optima marine batteries is used to provide the reserve power necessary for operation with engine off. Based on measurement of the sensor and computing load of 110A the 200AH battery bank will provide greater than one hour of operation. An additional benefit of the Xantrex unit is that when it is attached to an AC source (ie. shore power) it automatically switches the AC loads to the source and can supply up to 100A for charging the battery bank.

2.5 Environmental

The computing hardware is located in a common environmental enclosure in the bed of the F250.

2.5.1 Shock Isolation

The environmental enclosure is supported on each corner using Lord Heavy Duty Plateform shock isolation mounts. The selection of the mounts was based on analysis of the weight of the sled and components and a dynamics analysis for the F250 based on data collected using the INS system driving the F250 over the Grand Challenge 1 course.

2.5.2 Temperature and Dust

The environmental enclosure is cooled using a stock Ford Excursion auxiliary air conditioning unit mounted in the truck bed. This unit was selected based on an analysis of the thermal load of the computing cluster in conjunction with a solar load at 100 degrees Fahrenheit. The air is circulated in a closed loop with the environmental enclosure and the airflow design provides positive pressure within the enclosure to prevent dust from harming the computing components.

3 Autonomous Operation

3.1 Processing

3.1.1 High Level

The high level processing cluster is composed of 4 Fortress 2100 platforms using dual Opteron 250 processors running at 2.4GHz. Each dual processor unit has 2GB of RAM and a 6GB solid state disk for storage. This configuration and processor set was selected for both processing horsepower as well as ease of use and operation under a standard linux distribution. The high level processing cluster is built around the Suse 9.2 linux distribution running a 64-bit 2.6 kernel.

All processing in the cluster was designed and implemented using C and C++ in linux. The processes responsible for sensor and navigation processing as well as the high level path planning are statically allocated across the processing cluster. The interprocess communication is handled using the Neurtral Message Language (NML). This approach provides a common memory-mapped interface both within each and across all of the processors in the cluster. The interface to the low level control computer is handled with a CAN interface.

3.1.2 Low Level

The low level control computer is a dSpace Autobox and is responsible for the control interface between the high level processing cluster and the actuation elements. The low level control unit is implemented using a dSpace modular rapid control prototyping system. This system is equipped with a 800 MHz PowerPC process, 4 CAN interfaces, a MIL-STD-1553 interface, 32 channel A/D, 40 channels of timing and digital I/O, and 5 serial ports.

3.2 Hardware Configuration

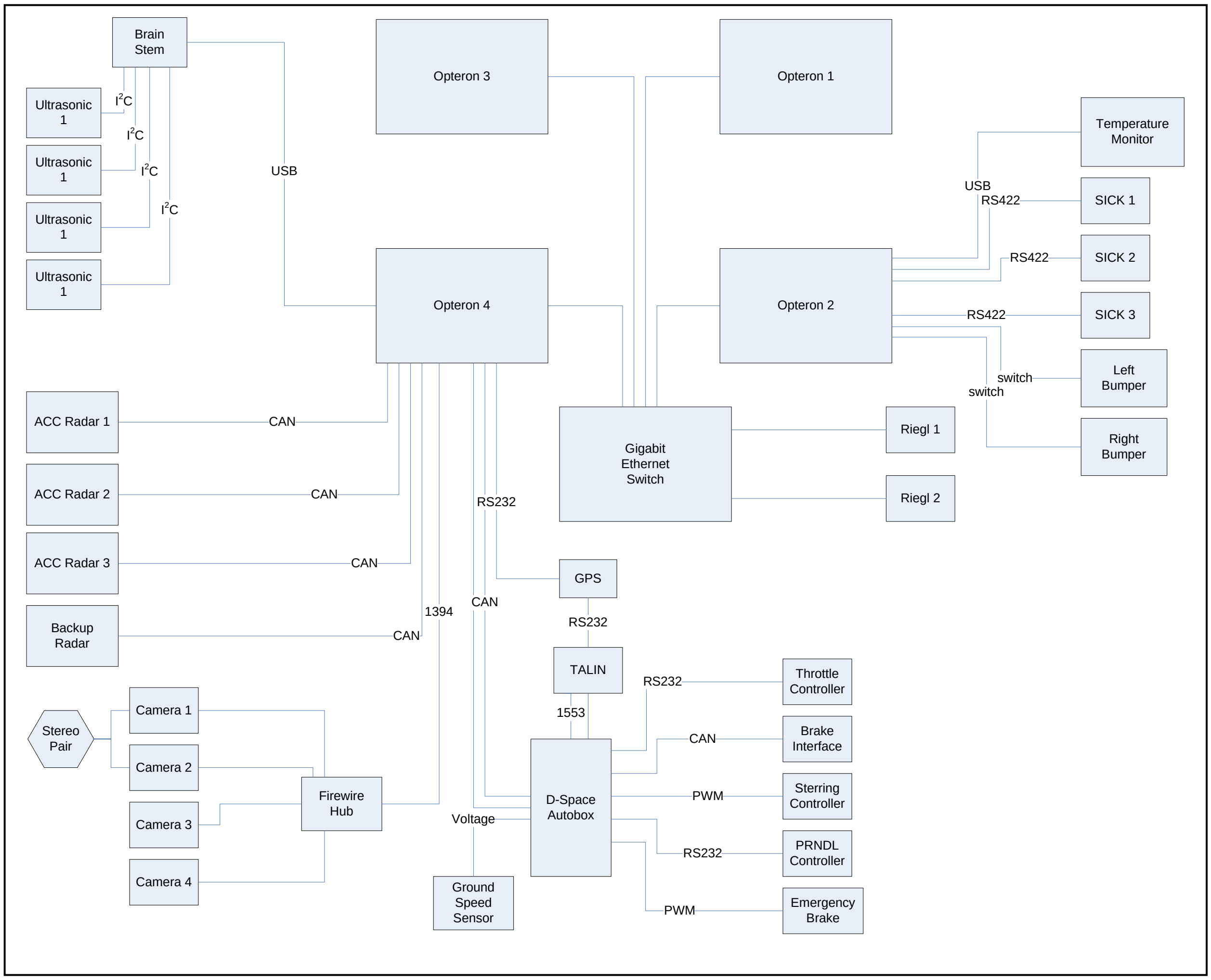


Figure 2. Hardware Configuration.

3.3 Localization

The navigation solution for all onboard components is provided by a blended navigation solution from a TALIN 5000 Inertial Navigator with precision satellite-corrected differential GPS and true groundspeed sensor inputs. The blended solution is provided across a MIL-STD-1553 interface from the TALIN to the low level dSpace computer.

3.3.1 DGPS

A NovAtel ProPAK-LBplus unit with OmniSTAR HP satellite-corrected differential GPS provides absolute position information at a 20 Hz update rate.

3.3.2 Inertial Navigation System (INS)

A Honeywell TALIN 5000 Inertial Navigator Unit provides a blended navigation solution at a 50Hz update rate. This solution incorporates DGPS data from the NovAtel unit, true groundspeed from a Vansco Doppler radar sensor, and internal acceleration and rotation data from precision accelerometers and a 3-axis ring laser gyro subsystem combined in an internal kalman filter.

3.3.3 Component and Blended Performance

The performance of the blended navigation solution is defined by the accuracy of the two navigation data sources. The best absolute navigation accuracy is provided using the OmniSTAR satellite-corrected DGPS with 10 cm CEP in the best case. On the other end of the spectrum, if no GPS is available the TALIN INS will provide position accuracy to 0.10% of distance traveled. The kalman filter internal to the TALIN provides a smooth navigation solution which incorporates a weighted combination of the absolute and relative sources based on GPS availability. The TALIN 5000 provides an excellent bridge for GPS-deprived areas with error on the order of one meter over one kilometer, greatly exceeding the expected duration of GPS outage.

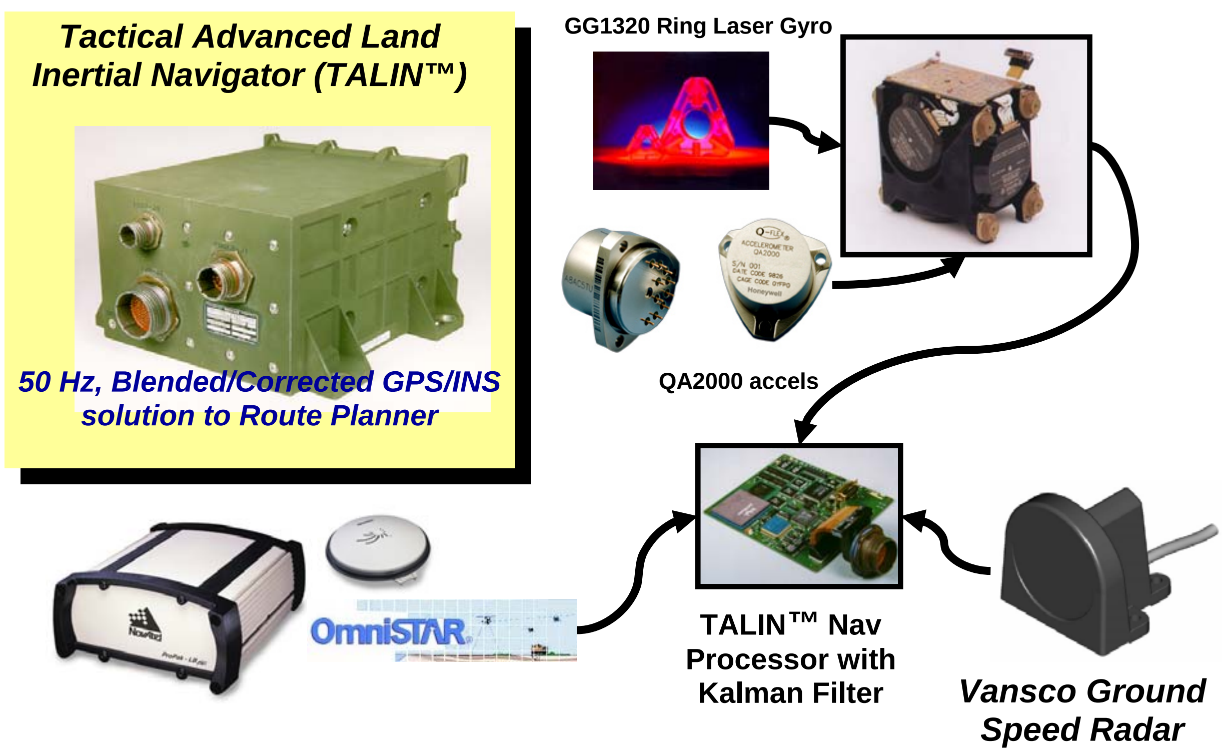


Figure 3. Navigation Subsystem.

3.4 Sensors

3.4.1 LADAR

Riegl Q120 Line Scan LADAR

Two Riegl Q120 LADAR units (80 deg at 0.4 deg/pixel with 100Hz scan rate) are mounted in the roof sensor suite with 10 and 30 meter lookahead to provide dense range information for terrain traversability assessment and construction of fused maps for path planning.

SICK LMS Line Scan LADAR

A combination of SICK LMS line scan ladar units are used for close range terrain sensing on the IVST1 vehicle. The LMS 290-S14 (90 deg at 0.5 deg/pixel with 75Hz scan rate) mounted in the roof sensor suite is used to provide a vertical slice of the terrain in front of the vehicle for assessment of the support surface, especially in undulating terrain. The LMS 211-30106 (180 deg at 1 deg/pixel with 75Hz scan rate) mounted on the front sensor rack is used both for profile following and hazard detection.

3.4.2 RADAR

Delphi Forewarn ACC3

The IVST1 vehicle is equipped with three production Delphi Forewarn® Smart Cruise Control radars. While a single radar is traditionally used to help reduce the need for drivers to manually adjust speed, manually apply brakes or disengage cruise control when encountering slower or stopping traffic, three sensors are mounted to the front of the IVST1 platform to provide enhanced long range object detection.

Smart Cruise Control radars use a mechanically-scanning, 76 GHz FMCW, long range radar sensor to detect objects in the vehicle’s path up to 500 feet (152 meters) ahead. The SCC radars have a 15 degree field of view. For the IVST, three SCC radars are configured across the front of the vehicle in a manner to provide a 45 degree field of view. This configuration is critical in order to detect potential obstacles while the vehicle is moving around blind corners. Data from the three radars is transmitted via a CAN interface to the Opteron cluster where it is fused with data from other sensors. An additional feature of the Smart Cruise Control radar is that it has an integrated yaw rate sensor. This sensor can act as a redundant sensor for the IVST1 vehicle.

Features

- Mechanically-scanning, 76 GHz FMCW radar-based sensing provides best-in-class performance in crowded, multi-target situations
- Integrated yaw rate sensor
- Sensor capable of being hidden behind front grille or fascia
- Automatic blocked sensor message
- Adjustable system sensitivity
- Self-alignment – simplifies service requirements
- Operates under a wide range of environmental conditions (dirt, ice, daylight, darkness, rain, and fog)

a) Delphi Forewarn ACC3

b) Delphi Dual Beam BUA

Figure 4. RADAR Sensors.

Delphi Dual-Beam BUA

The IVST1 vehicle is equipped with a production Delphi Forewarn® Dual-beam Radar Back-up Aid sensor. While designed for light-, medium- and heavy-duty vehicles to help drivers back up and park with more confidence, the IVST1 vehicle is using this sensor for short range detection of obstacles to the rear of the vehicle. This detection capability is important for those cases where a frontal obstacle requires the IVST1 vehicle to move backwards along an unknown path. The Back-up Aid's 24 GHz, dual-beam radar monitors up to a 6-meter (20-foot) deep by 2.75-meter (9-foot) wide area behind the vehicle. Using a CAN interface, obstacle data is transmitted to the Opteron cluster for threat processing.

Features

- One-piece radar/controller w/single connector and separate
- Continuous wave (non-Doppler) radar sensing detects moving and non-moving objects
- Up to 20-foot (6-meter) deep by 9-foot (2.75-meter) wide detection zone
- 24 GHz UWB US radio frequency compliance

- Operates under a wide range of environmental conditions (rain, snow, ice, fog, day, night, and noisy)
- Effective rear sensing range for both slow and faster backing maneuvers
- Back-Up Aid is automatically enabled when the vehicle is in reverse

3.4.3 Stereo

A fixed baseline stereo pair of firewire color cameras (Sony DFW-VL500) is located on the front sensor rack to provide dense local range data. The low level stereo processing is performed using the Small Vision System (SVS) software from Videre Design.

3.4.4 Ultrasonic Sensors

Ultrasonic sensors are present at the four corners of the vehicle covering the left and right sides at close range for tunnel following. These sensors are designed to provide real-time sensor feedback to maintain precise vehicle steering in these narrow corridors.

3.4.5 Bumper

An active bumper is integrated into the front sensor rack to provide a last line of response for protection of vehicle hardware and a means to recover from sensing and planning errors that could place the vehicle in contact with obstacles. The spring-loaded bumper provides discrete left and right contact signals to assist in refining the location of the bumper hit.

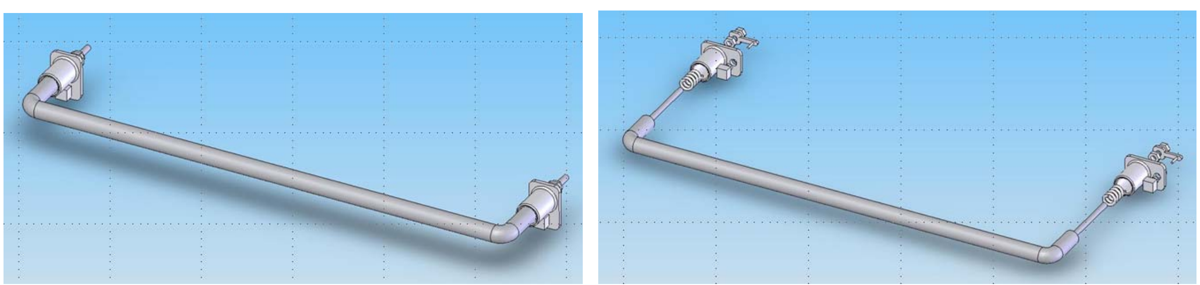


Figure 5. Active touch bumper.

3.5 Sensor Position and Envelopes

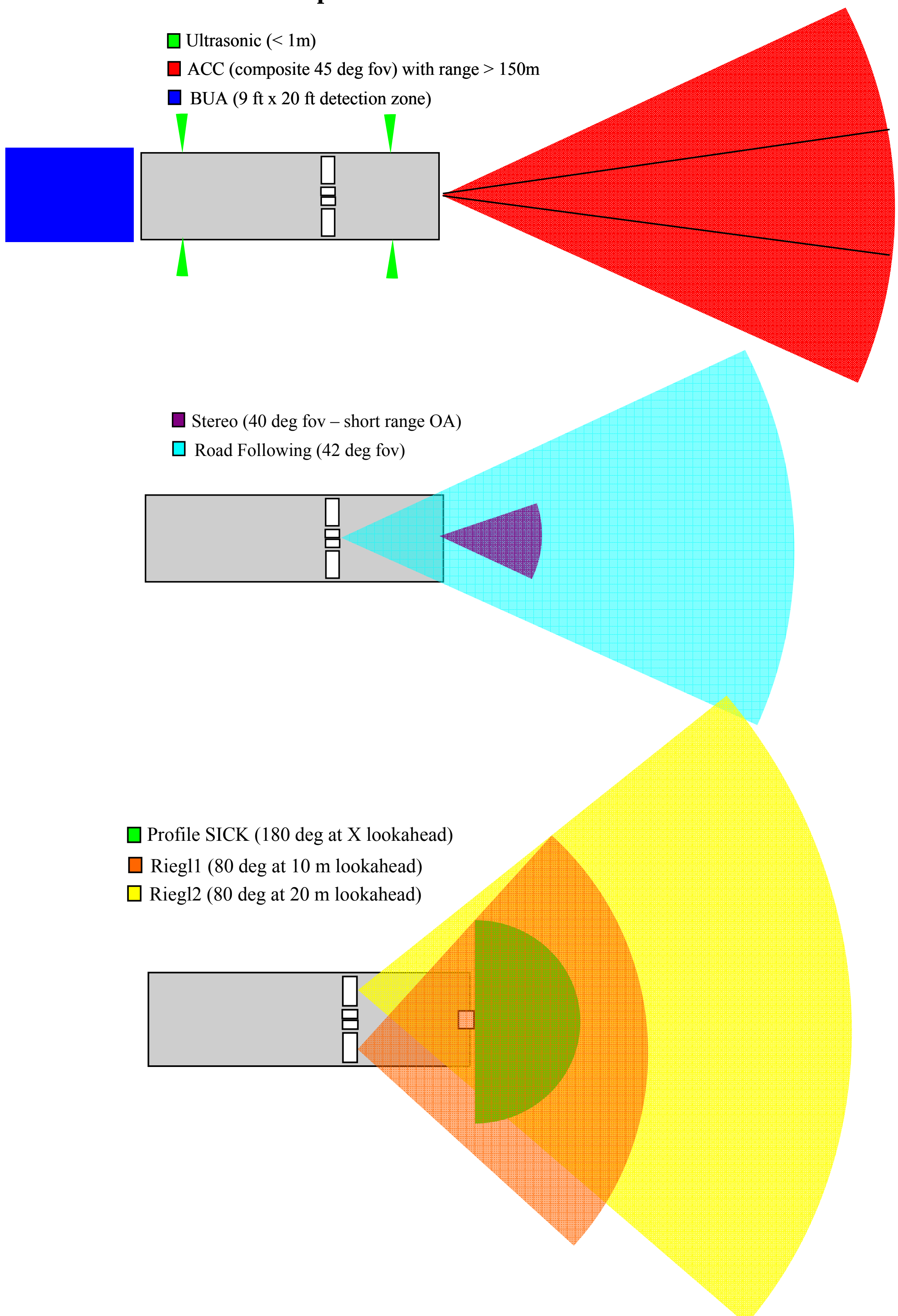


Figure 6. Sensor position and envelopes.

Figure 6 shows the general layout of the sensor components and their associated fields-of-view.

3.6 Control/Software Architecture

The IVST software architecture is based on the premise that not all sensors and algorithms are optimal for all environmental situations. For instance, an autonomous vision-based road-follower will not work well in an off-road situation where there is no road. It is even possible that if the algorithm is allowed to influence vehicle control in this situation, it can generate the incorrect response for the situation leading to navigation failure. Our software architecture design philosophy is to be able to dynamically and optimally reconfigure the software architecture based on the current environmental situation. We have created environmental contexts which represent categories of environmental situations that require a unique set of sensors to be active to optimally negotiate the specific environment context. Our environmental context set has evolved through our testing process in desert terrain. It is also simple to add additional environmental contexts if different behavior is required in a novel environment. Our set of environmental contexts includes:

- On-Trail-Straight – narrow trails with some, but little curvature and undulation
- On-Trail-High-Curvature – narrow trails with high horizontal curvature
- On-Trail-Undulating – narrow trails with highly periodic undulation
- Structured-Road – paved and lined roadway
- Paved-Road – paved, but no lane demarcations
- Wide-Road – very wide and fairly straight roadway
- Dry Lakebed – wide open and flat terrain
- Tunnel – the course goes through a tunnel or underpass
- Off-Road – complex, highly foliaged and rock strewn terrain off of the road
- Qualifier – NQE environment
- Unknown – default context when a context determination cannot be made.

With each environmental context, there is an associated set of algorithms tied to specific sensors. The Unknown context serves as a conservative mode of operation that theoretically should be able to function in any of the environmental contexts but at slower speeds and less navigation efficiency. The configuration for each context is file configurable which has allowed for optimal determination of sensors and algorithms per context through environmental testing.

The use of environmental contexts generates specific software architecture requirements. First, the system needs to be able to automatically determine its environmental context. Once the context has been determined, the architecture needs to be able to dynamically reconfigure for that specific context. And finally, the software architecture must be able to execute the behavior in the context. Our software architecture is shown in Figure 7.

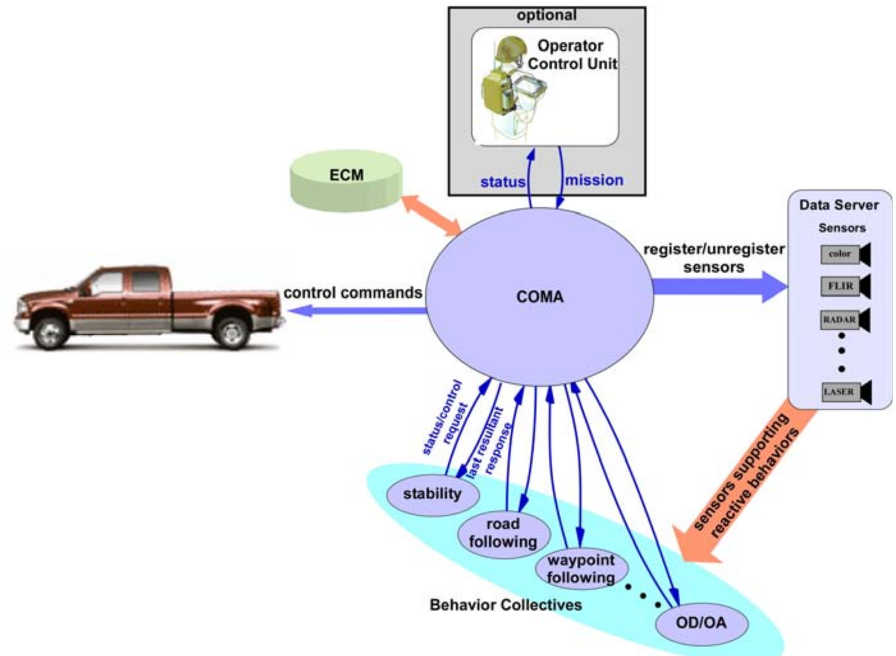


Figure 7: IVST Software Architecture

In order to estimate current environmental context, we have developed a module called the Environmental Context Module (ECM). Its main responsibility is to determine a current context based on all available information. The sources of information that are used by ECM are the attributes of the RDDF such as waypoint speed, tolerance, density and curvature, inputs from perception based algorithms, and data from the inertial measurement unit (IMU). ECM is essentially a classifier and is trained in a supervisory fashion to recognize environmental contexts by exposing the algorithm to the individual set of contexts and allowing the algorithm to develop a mapping between the various input sources and the specific environmental context. It is also possible in ECM to manually add to the mapping if a known relation exists between input attributes and a specific environmental context. At runtime, the ECM gathers all the information from the various input sources and produces an estimate of the current environmental context based on its internal learned mapping along with a confidence measure.

Once the environmental context has been determined our software architecture must dynamically reconfigure itself for that specific context. At the center of our architecture is a module called COMA (Contextual Operating Mode Architecture). COMA is a hybrid robot architecture which combines both reactive and deliberative architectural qualities into one architecture. At the lowest level of COMA is the reactive layer of behaviors. Behaviors are categorized into positive and negative behaviors. A positive behavior is one that has a specific direction it desires to travel. A road-follower is an example of a positive behavior because it is attempting to steer the vehicle to the center of a road. A negative behavior is one that blocks candidate steering directions based on the hazard level of those candidates. The obstacle avoider is an example of a negative behavior. An arbiter is used to fuse the control responses into one resultant signal that is passed to the vehicle controller for execution. The set of behaviors includes:

- Waypoint-following – a positive behavior that controls vehicle steering and speed to follow a set of predetermined waypoints and keep the vehicle within a prescribed corridor
- Obstacle Avoidance – a negative behavior that receives sensor detections, places them in a vehicle centered map and analyzes the costs of candidate steering directions based on their proximity to detected hazards in the map. The obstacle avoidance algorithm can accept inputs from radar, laser, stereo and mechanical active bumper. The obstacle avoider is considered a "negative" behavior
- Road-following – a positive behavior that attempts to steer the vehicle to the center of the road
- Profile Following – a positive/negative behavior that uses the geometric road boundaries such as burms and dropoffs to control the vehicle.
- Drop-off – a negative behavior that only controls speed and immediately stops the vehicle on the detection of a drop-off in front of the vehicle
- Tunnel-Follower – a positive behavior that uses inputs from acoustic sensors mounted at the four corners of the vehicle to steer the vehicle through a tunnel
- Proprioceptive Control – a positive behavior that control the vehicle speed based on terrain roughness

- Retrotraverse – a problem resolution behavior that activates when the other components of the system determine there is no forward traversable path. Retrotraverse backs the vehicle along its previously recorded path.

The arbiter combines the outputs from the current set of contextual behaviors to produce a resultant steering and speed response. Each positive behavior sends its desired steering response to the arbiter in the form of a vector that represents the full range of steering, with the value at each vector element being the degree to which that specific behavior believes the vehicle should steer. Negative behaviors send a response over the full steering range that represents steering directions not to go. The arbiter produces a weighted sum of all positive behaviors where the weight of a behavior is the produce of an assigned relative weight of the behavior to other behaviors in a specific environmental context and the confidence of the behavior. The superposition of the sum of negative behaviors is used to cancel out hazardous directions and then the peak of the final response is used as the desired steering direction. Those behaviors that control speed, will also provide a speed vector over the full steering range, where the value of a vector element represents the speed the behavior wants to go for that steering direction. The arbiter takes the minimum speed over all behaviors for the chosen steering direction.

3.6.1 Waypoint Following

The waypoint following behavior is the backbone to our architecture as it provides the general path through the terrain to a goal point. The waypoint follower does not take into account perceptual information; its main purpose is to minimize the vehicle's normal distance from the line segments connecting adjacent waypoints which come from an external source. The waypoint follower produces a desired steering and speed signal; the speed computation is based on the density of waypoints and the geometry they carve out. The waypoint list is provided to the waypoint follower either through an Operator Control Unit or by downloading the list from a file. The waypoints can be in either Lat/Long or UTM coordinates.

The waypoint follower generates steering arcs based on a look-ahead along the current executing path segment (Figure 8). The speed set point is generated based on the desired steering radius using the following relationship

$$Speed\_set\_point = desired\_speed * f \quad (1)$$

Where f is a multiplying factor such that $0 \le f \le 1.0$. Equation 1 ensures that the speed set point generated by the algorithm never exceeds the desired speed.

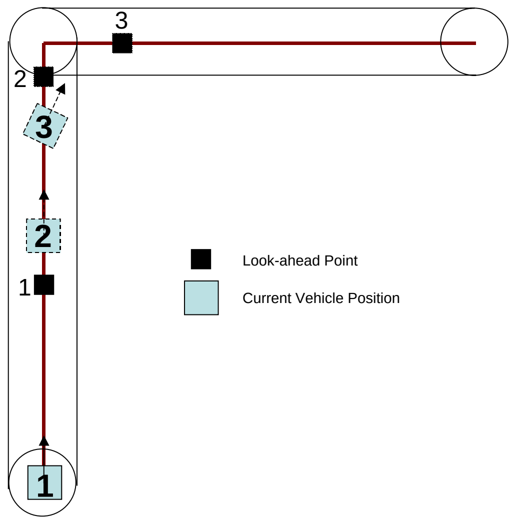


Figure 8: Pictorial depiction of the look-ahead point generation.

The steering radius (R) is computed by

- Finding the closest point (C) on the current executing path segment with respect to the current vehicle position (χ).
- Computing the look-ahead point (L). L is *look-ahead* distance from C along the given path segment(s).
- Defining V as the vector between χ and L, that is, $V = L - \chi$.
- Rotating V from the inertial coordinate system to the body fixed coordinate system, that is:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos(\psi) & \sin(\psi) \\ -\sin(\psi) & \cos(\psi) \end{bmatrix} \mathrm{V} \quad (2)$$

- Finally, R is:

$$R = \frac{x^2 + y^2}{2y} \quad (3)$$

The variable ψ in Equation 2 is the vehicle heading. The steering arc generation is pictorially shown in Figure 9.

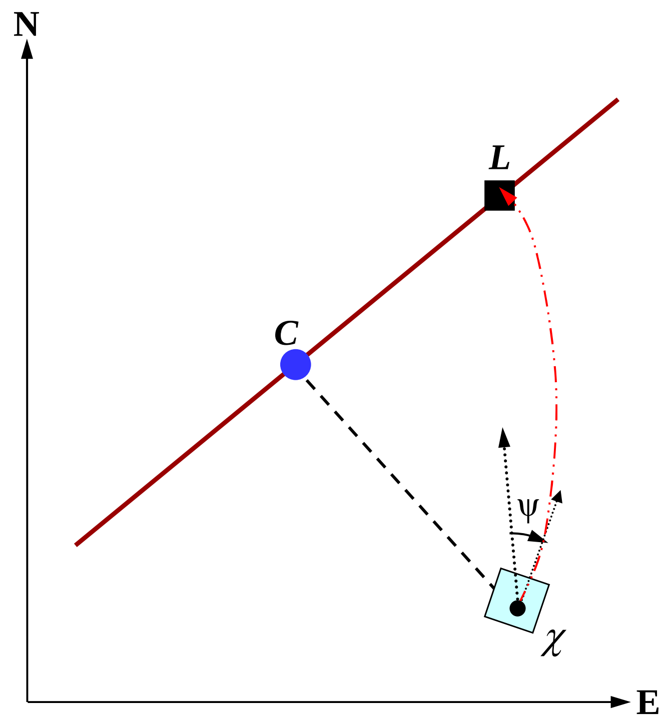


Figure 9: Generation of the steering radius *R*.

3.6.2 Profile Following

The profile follower uses a laser scanner to determine the position of the geometric edge of the road and generates steering arcs to steer the vehicle away from it. This behavior accurately measures the distance from the front of the vehicle to each point on a line going across the road to determine the extents of the navigable portion of the road, including the position of the road edge, and whether it falls away or ends in a vertical wall.

Having obtained a sequence of measurements of the height of the road in front of the vehicle, the behavior determines if there is a sharp edge to the road by computing the variance within a window that moves over the obtained data. If the data (heights of the road) are denoted by z_i, then the mean over the N data points centered on point i is

$$\bar{z}_i = \frac{1}{N} \sum_{j=-(N-1)/2}^{(N-1)/2} z_{i+j} \tag{1}$$

and the variance is

$$V_i = \frac{1}{N} \sum_{j=-(N-1)/2}^{(N-1)/2} (z_{i+j} - \bar{z}_i)^2 \tag{2}$$

Typically, we take N, the size of the moving window, to be 5.

If the road is smooth and navigable, the variance will be low, indicating that the heights of the points on the road are relatively constant. However, if there is a sharp drop-off at the edge of the road, the data will exhibit high variance at the road edges, since the distance between the points on the road and the points off the edge will be large. Similarly, if there is a wall at the edge of the road, the points on the wall will be far above the road points, so the variance will be high. The behavior thresholds the variance of the data to obtain the road-edge position. Figures 10 and 11 show the sensor output and the corresponding variance plot of the data respectively. Note that the sharp peak at the right in Figure 11 indicates a steep drop-off.

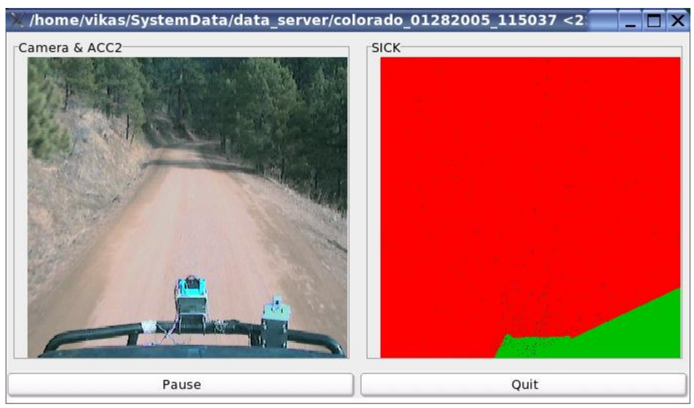


Figure 10: Profile follower and the corresponding Laser scanner output. The image on the right is an upside-down cross-section of road (red is empty space). Visible are the steep bank on the left, the ditch, the flat road, and the drop-off on the right.

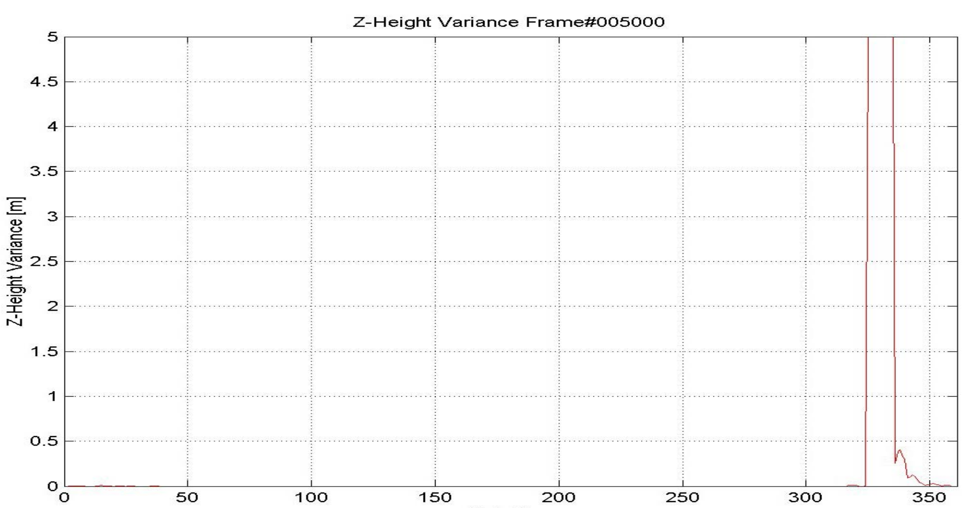
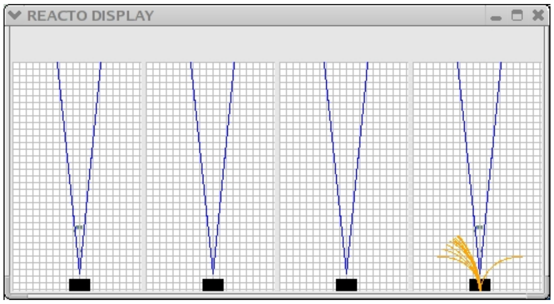


Figure 11: Variance plot of the road as shown in Figure 10.

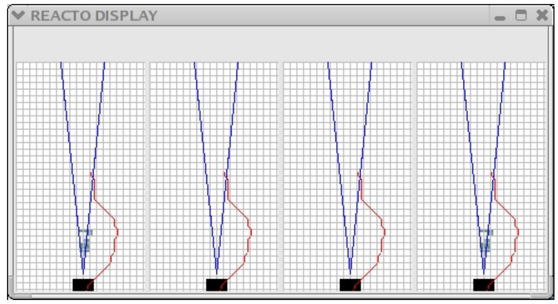
3.6.3 Obstacle Detection and Avoidance

The IVST obstacle avoidance behavior is called REACTO (REACTive Obstacle navigation). REACTO can accept inputs from any combination of sensors with the requirement that the sensor data is tagged with 3D position information if this is not native to the product of the sensor (such as raw color data produced by a color camera). Through the course of this project, REACTO has been interfaced with a Delphi ACC3 Radar, a SICK laser scanner and a Riegl laser scanner. In addition to interfacing with sensors, REACTO can also accept inputs from sensor pre-processing modules, such as the Artificial Immune System scene segmentor that "labels" the terrain into a set of terrain type categories or as traversable and non-traversable regions. REACTO maintains a vehicle-centered, vertically layered, 3D cell-based spatial map. A vehicle-centered map is different than an absolute map in the sense that the vehicle stays in the same place in the map and the prior map data translates with respect to vehicle motion rather than the vehicle moving amongst fixed data as in an absolute map. REACTO has three layers in its map, and the layers correspond to different relevant dimensions of the vehicle. The topmost layer of the map corresponds to overhead clearance of the vehicle, i.e., if a detection occurs in this layer, the vehicle will be able to go under the detected feature. An example of this is a "canopy" created by the branches and leaves of a tree. The middle layer corresponds to the vertical dimension of the vehicle body and a non-traversable detection in this layer would warrant a planning response to go around the hazardous detection. The lowest layer in the map corresponds to the vehicle ground clearance and the vehicle's ability to cross a negative obstacle such as a hole or rut.

From the vehicle-centered model of the world, REACTO must determine the appropriate steering response in order to avoid hazards to the vehicle. REACTO can use either of two mechanisms to accomplish this. The first method is based on evaluating a set of candidate steering arcs. The evaluation of the arcs is based on where the arcs pass through the internal map model of the world and where the goal point is relative to the vehicle; see Figure 12(a). Steering arcs that pass through hazards will score zero and steering arcs that have adequate lateral vehicle clearance will score high. In addition, steering arcs that bring the vehicle closer to the goal point are scored higher than those steering arcs that take the vehicle further from the goal point. REACTO will also produce a maximum speed for each steering arc based on the estimated



(a) arc based



(b) astar planning

Figure 12: REACTO steering generation with (a) arc based evaluation, (b) A* path generation.

stopping distance of the vehicle from its current speed measurement. In the steering arc mode, REACTO acts as a negative behavior used in conjunction with a waypoint follower.

The second method of obstacle avoidance planning uses an optimal planning technique called A* search. A* search is a constraint based planner that uses heuristics to produce an optimal plan or route from the vehicle's current location to a goal point provided by the waypoint follower, see Figure 12(b). The constraints used in our A* planner are the minimum distance constraint and a terrain traversibility constraint. In the A* search planning mode, REACTO provides a positive control command to the behavior arbiter. In the A* search planning mode, if a plan cannot be built, REACTO will temporarily revert to the steering arc mode until it can successfully produce an A* plan. We run the A* version of REACTO with a waypoint follower.

3.6.4 Sensor Fusion

The IVST system has a host of perception based sensors from lasers, to color cameras to radars. One component of our software architecture is our data server. The data server provides a consistent interface for the algorithms to grab sensor data no matter what processor the process resides or what processor directly reads the sensor data. Any algorithm can access data from any number of sensors. In order to reduce sensor fusion complexity, we have essentially tied one perception sensor to one instance of a behavior. Therefore the sensor fusion problem is handled indirectly through the fusion of behaviors in the arbiter. This one-to-one sensor to algorithm mapping also plays in well with our environmental context based architecture where certain sensors may not perform well in certain environments.

3.7 Process Allocation

Figure 13 shows the allocation of the different component processes to the four Fortress 2100 dual Opteron machines. This allocation is statically assigned and each machine has an initialization script that runs at boot to configure operation.

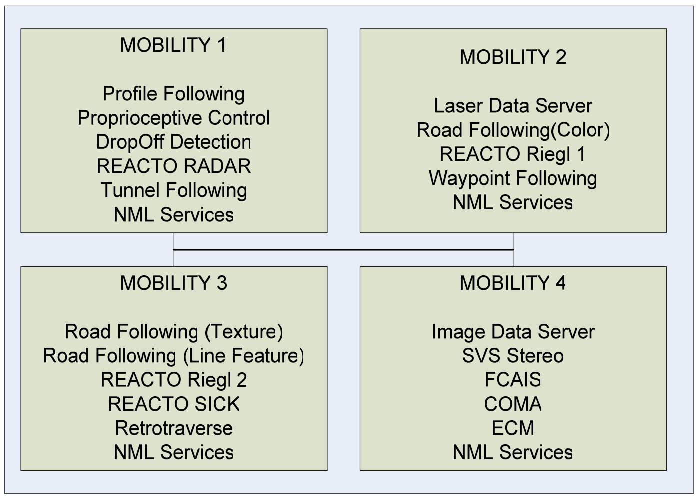


Figure 13. Process Allocation.

4 System Testing

4.1 Data Collection on Grand Challenge 1 Course

In October 2004 an initial survey of the Grand Challenge 1 course was performed and in December 2004 a team of engineers with two sensor instrumented platforms drove large segments of the course, collecting navigation, image, and laser data for algorithm development and design validation for components such as the shock isolation sled.

4.2 Ranch Testing

The team has used ranch facilities near Sedalia and Limon, Colorado to perform both algorithm development and initial system testing for each of the environmental contexts on the F250. While this site does not have environmental characteristics identical to the desert, the area has roads, trails, and rugged terrain for validation and these areas are sufficiently similar to prove out the value of new approaches as well as to perform extended runs over 50 kilometers.

4.3 Desert Test I

August 3-13, 2005 the team performed testing with the fully configured F250 platform at sites northeast of Las Vegas, NV. This testing concentrated on

- Validation of the platform in extreme environmental conditions (heat, sun, dust) exceeding those expected at the race,
- Training of algorithmic components in the target environment,
- Validation of sensor performance over the full dawn-to-dusk range in the target environment, and
- Extended runs (>10 miles) using RDDF files with evaluation of performance of perception, planning, and control using the environmental context paradigm.

This validation process in the desert environment, while grueling, provides both critical data for improving performance as well as positive confirmation for the good design choices.

4.4 Desert Test II

August 29-September 5, 2005 the team will return to the same set of desert sites and continue validation testing. The focus in this testing will evaluate changes made in response to the first round of desert testing as well as provide increased focus on full run operation from RDDF to crossing the last waypoint.

4.5 Final Preparation

In early September a mock NQE will be constructed in Littleton to test system performance and refine the operational sequence to ensure that the team is ready to perform in the structured NQE environment and that each qualifying opportunity will be used. In the week prior to the NQE event a final set of validation testing will be performed at the desert test site end-to-end runs of RDDF courses using the GCE procedures.

5 IVST1 Team

The IVST1 Vehicle is a team effort and represents a substantial personal commitment by each member. The current team roster includes the following members:

Vikas Bahl
Ari Daum
Alan Griesbach
Floyd Henning
Jerry Ivan
Peter James
William Klarquist
James McBride
Jeremy Nett
Joseph Raad
Venkat Rajagopalan
Douglas Rhode
Mark Rosenblum
David Simon
Aakash Sinha
Joseph Stinnett
Alan Touchberry
Douglas Turner
Richard Weaver